**Listing of Claims:**

This listing of claims will replace all prior versions and listings of claims in the application:

1. (Previously presented): A method for identifying protocol encapsulation in received network data comprising providing language definition including a grammar; receiving incoming network data and processing said incoming network data in accordance with a formal language processing technique using said language definition, said processing including parsing said network data using said grammar, said network data being organized into data packets.

2. (Original): The method of claim 1 wherein said grammar is a grammar graph, the method further including providing a deterministic finite automaton (DFA) representing said grammar graph.

3. (Original): The method of claim 1 further including scanning said incoming network data using lexical token scanning to produce plural lexical tokens, said step of parsing including parsing said lexical tokens.

4. (Original): The method of claim 3 wherein said lexical scanning includes providing a set of regular expressions.

5. (Original): The method of claim 3 further including providing a deterministic finite automaton (DFA), said DFA including a representation of said lexical tokens and said grammar, said step of scanning including recognizing lexical tokens contained in said data packets using said DFA, said step of parsing including identifying grammatical structure among said lexical tokens using said DFA to identify protocol encapsulation in said incoming network data.

6. (Previously presented): In a data packet network switching device, a method for processing data packets comprising:

providing a language definition including a grammar;

receiving plural data packets, each having a length not necessarily equal to one another; and

for each data packet, processing said data packet according to a formal language processing technique using said language definition including lexically scanning said data packet to produce plural lexical tokens, parsing said lexical tokens using said grammar to produce one or more identified protocols, and processing said data packet based on said identified protocols.

7. (Original): The method of claim 6 further including compiling said grammar to produce a grammar graph.

8. (Original): The method of claim 7 wherein said lexical scanning includes providing regular expressions for identifying said lexical tokens.

9. (Original): The method of claim 8 further including compiling said regular expressions are into a deterministic finite automaton (DFA).

10. (Original): The method of claim 9 further including incorporating said grammar graph into said DFA.

11. (Previously presented): In a data packet receiving and forwarding device, a method for processing data packets comprising a stream of data, said method comprising:

receiving a description of grammar rules in the form of a grammar packet classification language;

compiling said grammar packet classification language to produce a grammar graph;

configuring a programmable grammatical packet classifier with said grammar graph;

processing said data stream in accordance with a formal language processing technique using said grammar packet classification language including parsing said data stream with said grammatical packet classifier to identify a protocol structure in a received data packet; and

processing said received data packet in accordance with said protocol structure.

12. (Original): The method of claim 11 further including:

receiving a description of classification rules in a lexical classification language;

compiling said classification language to produce a deterministic finite automaton (DFA) comprising plural states;

configuring said hardware packet classifier with said DFA; and

scanning said data stream with said hardware packet classifier to produce plural lexical tokens,

wherein said parsing is a step of parsing said lexical tokens.

13. (Original): The method of claim 12 wherein said grammar graph is incorporated into said DFA.

14. (Original): The method of claim 12 wherein said lexical classification language includes regular expressions.

15. (Original): The method of claim 14 wherein said regular expressions include arithmetic and logic operations.

16. (Original): The method of claim 15 wherein said regular expressions further include skip operations.

17. (Original): The method of claim 16 wherein said regular expressions further include data storage operations.

18. (Previously presented): A network data packet classifier comprising:

an input port for receiving network data packets comprising a stream of data;

a memory assemblage configured with data representing a deterministic finite automaton (DFA), said DFA defined by a language definition and representing a grammar graph and plural regular expressions; and

decompression logic operatively coupled to said memory assemblage and configured to process said stream of data according to a formal language processing technique using said language definition including a step to scan said stream of data with said DFA to find a matching one of said regular expressions thereby producing plural lexical tokens,

said decompression logic further configured to parse said lexical tokens with said DFA to identify a protocol structure in a received network data packet,

wherein processing of said network data packet depends on said protocol structure.

19. (Original): The classifier of claim 18 wherein some of said regular expressions include arithmetic instructions and logic instructions, said memory assemblage further configured to contain said instructions, the classifier further including an arithmetic logic unit operatively coupled to said decompression logic and configured to execute said instructions.

20. (Original): The classifier of claim 19 further including at least one register operatively coupled to said arithmetic logic unit, said arithmetic logic unit further configured to store data into said register in response to a save instruction.

21. (Original): The classifier of claim 19 further including skip logic operatively coupled to said logic component and configured to skip over an amount of data in response a skip instruction.

22. (Original): The classifier of claim 18 wherein said network data packets can vary from one packet to another.

23. (Original): The classifier of claim 18 wherein said DFA is in compressed form.

24. (Original): The classifier of claim 23 wherein said DFA comprises plural non-default states and plural default states, and said memory assemblage comprises a base memory, a next-state memory, and a default-state memory; said base memory configured to contain address locations of said next-state memory, said next-state memory representing all of said non-default states, said default-state memory representing all of said default states.

25. (Original): The classifier of claim 24 wherein said memories are random access memories.

26. (Original): The classifier of claim 24 wherein said memories are read-only memories.

27. (Previously presented): A network packet classifier comprising:

means for receiving an incoming network packet; and

means for identifying protocol structure in said network packet including means for processing said network packet in accordance with a formal language processing technique using a language definition, including a step of scanning to match patterns in its constituent data against plural regular expressions to produce lexical tokens and means for parsing through said lexical tokens using a grammar, said regular expressions and said grammar being defined by said language definition.

28. (Original): The classifier of claim 27 wherein said means for scanning includes a memory component configured with data to represent a deterministic finite automaton (DFA).

29. (Original): The classifier of claim 28 wherein said memory component is further configured to include said grammar.

30. (Original): The classifier of claim 27 wherein said regular expressions include arithmetic specifiers and said means for classifying includes an arithmetic logic unit configured to perform operations in accordance with said arithmetic specifiers.